PhotonFirst

API

User Manual for PhotonFirst API

v1.1.1

April 2023

Thank you for purchasing the PhotonFirst API product. This manual has been prepared for users of the PhotonFirst API. To ensure correct use, please read this manual carefully before using these products.

- Although every effort has been made to ensure the accuracy of this manual, if you note any points that are unclear or incorrect, contact PhotonFirst.
- Read the instruction manuals for any other products that you are using with this product (a computer or other peripheral equipment).
- If the product is used in a manner not specified by the manufacturer, the protection provided in the product may be impaired.

PhotonFirst Technologies B.V.	
Address	Pyrietstraat 2a 1812 SC Alkmaar The Netherlands +31 85 00 76 700
Email	team@photonfirst.com
Web	www.photonfirst.com
Sales support	team@photonfirst.com
Technical support	support@photonfirst.com
Publication date	April 2023
Document version	Version 1.1
Device type	Modular Photonics, SwitchedGator, Gator
DMR	PhotonFirst API DMR 1.1.1
SW version	PhotonFirst API v1.1.1

Information in this document is subject to change without notice.

Reproduction in any manner whatsoever without the written permission of PhotonFirst is strictly forbidden. Trademarks used in this text: PhotonFirst, the PhotonFirst logo, are registered trademarks of PhotonFirst. Other trademarks and trade names may be used in this document to refer to the entities claiming either the marks, names or their products. PhotonFirst disclaims any proprietary interest in trademarks and trade names other than its own.

Referenced documents

Ref.	Document name, source, reference, version, date
[1]	Modular Photonics 003 User manual, v1.0, October 2022
[2]	PhotonFirst SGTR User Manual, v3.5, May 2021
[3]	PhotonFirst GTR User Manual, v7.0, June 2021

1 Terms, definitions, abbreviations, and conventions

1.1 Terms and definitions

Term or definition	Description
Channel	Optical port on the interrogator to which a sensor array can be attached.
Sensor power	The amount of optical power reflected by an FBG sensor over a period specified in the acquisition rate.
F- Factor	Electrical amplification of photodiode signal in the system, which is controllable for the system. 8 is the maximum value, 125 is the lowest value.
Centre of Gravity value	Weighted Average of the sensor power for a specific sensor
Centre of Gravity wavelength	Centre of Gravity value converted to wavelength based on calibration of the interrogator device
Sensor	A single FBG sensor within a sensor array
Sensor array	Fiber sensor containing one or more FBG sensors
Wavelength precision	The random spread of measured values around the average measured values. Closeness of agreement between indications or measured quantity values obtained by replicate measurements on the same or similar objects under specified conditions. The wavelength precision is also referred to as noise
Wavelength measurement range	The measurement wavelength interval of the device. Although the wavelength range can differ between devices, the wavelength range in the specifications is the one guaranteed for all devices.
Wavelength repeatability	The measurement precision of a unit under test when measuring under a reproducible set of conditions over a short period of time. These conditions include the same measurement procedure, same operators, same measuring system, same operating conditions and same location and a similar unit under test. Wavelength repeatability is also referred to as drift
Wavelength resolution	Smallest change in a quantity being measured that causes a perceptible change in the corresponding indication. The smallest to be distinguished magnitude of a wavelength value. Resolution of MP family systems is the minimal Centre of Gravity wavelength step size.
Sensor centre wavelength	The wavelength value where the reflectivity peak of an FBG sensor is located when the sensor is at rest. This is one of the parameters to which an FBG sensor is manufactured.

Term or definition	Description
Sensor dynamic wavelength range	The wavelength measurement range allocated for an FBG sensor in the wavelength range of the device. The extremes of the measurement range are the minimal and maximum wavelength expected for an FBG sensor.
Sensor amplitude dynamic range	The allowed range in peak reflectivity for all FBG sensors interrogated by a device. The range should be valid for the entire dynamic wavelength range allocated to a FBG sensor. The dynamic amplitude range is dependent on the sensitivity setting of a device.
Sensor spacing	The difference between the center wavelength value of two neighboring FBG sensors.
Sensor peak reflectivity	Reflectivity of a FBG sensor is its effectiveness in reflecting radiant energy. It is the fraction of incident electromagnetic power that is reflected at an interface. The reflectivity of an FBG sensor is specified for the center wavelength
Sensor bandwidth	Bandwidth of the FBG sensor where reflected intensity of the FBG sensor is -3 dB of the FBG center wavelength peak intensity
Side lobe suppression ratio (apodization)	The Side Lobe Suppression Ratio (SLSR) is the ratio of intensity difference between the FBG center wavelength peak and the biggest side lobe. SLSR is can also be called apodization
Polarization dispersion	FBG writing induced birefringence effect, causing splitting of the FBG spectrum into two polarization axes.
Polarization amplitude dependency	The dependence of the intensity of the FBG center wavelength peak on the polarization.
Sensitivity setting	Device setting which influences the sensitivity of the device. Depending on the device, the sensitivity settings is applicable for a single or for all channels.
Sample rate	Frequency interval at which the device takes measurements, inverse of the measurement integration time
Differential behavior of neighboring sensors	The behavior of neighboring sensors in relation to each other. I.e. the expected difference CoG wavelength values in relation to each other.

1.2 Abbreviations

Abbreviation	Meaning
P1	PhotonFirst
ADC	Analog Digital Converter
API	Application Programming Interface
CoG	Centre of Gravity
CSV	Comma-Separated Values

Abbreviation	Meaning
CTE	Coefficient of Thermal Expansion
FBG	Fiber Bragg Grating
FC/APC	Fiber Optic Connector for Angled Physical Contact
FWHM	Full Width at Half Maximum
FTDI	Future Technology Devices International
MOS	Modular Photonics Operator Software
GOS	Gator Operator Software
LED	Light Emitting Diode
SMF	Single mode fiber
USB	Universal Serial Bus
SLRS	Side Lobe Suppression Ratio
RoHS	Restriction of Hazardous Substances
PMD	Polarization Mode Dispersion
COTS	Commercial Of the Shelf Software

1.3 Reading conventions

This section gives an overview of the stylistic and syntax conventions and their meaning.

DANGER ‘DANGER’ indicates a dangerous situation which, when the safety instructions are not followed, will result in serious or deadly injuries

WARNING ‘WARNING’ indicates a dangerous situation which, when the safety instructions are not followed, can result in serious or deadly injuries and/or serious damage to the product or its environment

CAUTION ‘CAUTION’ indicates a situation which, when the safety instructions are not followed, can result in serious or minor to average injuries and/or damage to the product or its environment

REMARK ‘REMARK’ highlights important information for the user not related to injuries

Table of contents

2 Introduction

This document describes how to install, configure and operate the PhotonFirst API. The PhotonFirst Python API will allow interaction with the following products:

- MIP-003
- SGTR
- GTR

2.1 Intended use

The PhotonFirst Python API is to be used to control and readout PhotonFirst interrogator systems via an accessible programming platform. It allows for integration of the PhotonFirst interrogator into the dedicated environment of the user.

PhotonFirst interrogators are intended for general use in fiber-optic sensing and testing. Although the internal hardware is capable of withstanding harsh environments, the products in their form factor should be operated in an environment which meets the environmental conditions specified in the product specification as described in the user manual of the applicable interrogator (see referenced documents).

2.2 Intended users

The PhotonFirst API should only be used by competent users, which are familiar with building software & hardware applications. It is expected that the user is well informed on the use, handling and safety aspects regarding optical fibers and interrogators. A non-exhaustive list of precaution and safety aspects during usage can be found in the user manual of the applicable interrogator (see referenced documents).

2.3 Supported platform

The PhotonFirst API will provide support for Windows 10 or higher. The supported Python version is 3.8 or higher.

2.4 Software deliverables

Software	Vendor	Version
PhotonFirst API	PhotonFirst	1.1.1
PhotonFirst API – Sample application	PhotonFirst	1.1.1
Python	COTS (Python Software Foundation)	3.8.10
Bonjour	COTS (Apple Inc)	2.0.2.0
Microsoft Visual C++ 2015 Redistributable	COTS (Microsoft Corporation)	14.34.31938

3 Software Installation

This section describes how the software should be installed and configured prior to operation.

3.1 COTS software

3.1.1 Python

1. Run the Python installer (`python-3.8.10-amd64.exe`).

   The installer wizard will now be opened.

2. Check: `Add Python 3.8 to PATH`
3. Click `Customize installation`
4. Click `Next`
5. Check: `Install for all users`
6. Click `Install`
7. When asked “Do you want to allow this app to make changes to your device”, click `Yes`.

   Python will now be installed.

8. Click `Finish`

3.1.2 Bonjour

1. Run the Bonjour installer (`BonjourPSSetup.exe`).

   The installer wizard will now be opened.

2. Verify that the software version in the lower right corner states:
   `2.0.2.0 x64`
3. Click `Next`
4. Select “I accept the terms in the license agreement” and click `Next`
5. Click `Next`
6. Uncheck `Create Bonjour Printer Wizard desktop shortcut`
7. Uncheck `Automatically update Bonjour Print Services and other Apple software`
8. Click `Install`
9. When asked “Do you want to allow this app to make changes to your device”, click `Yes`.

   The BonjourPS service will now be installed.

10. Click `Finish`

3.1.3 Microsoft Visual C++ 2015 Redistributable

1. Run the VC redistributable installer (`vc_redist.x64.exe`).
   Verify that the software version in the upper right corner states:
   `14.34.31938`
2. Select “I agree to the license terms and conditions” and click `Install`
3. When asked “Do you want to allow this app to make changes to your device”, click `Yes`.
4. Click `Close`

3.2 PhotonFirst software

This section describes how to install the PhotonFirst API and the sample applications.

3.2.1 PhotonFirst API

1. Click the Start menu button in windows and type File explorer. Click File explorer when found.

2. Navigate to: C:\Users\YOUR USERNAME\Desktop

3. Create a new folder named “PhotonFirst” and copy the Python wheel files (.whl) and sample applications to this folder.

4. Click the Start menu button in windows and type Powershell
5. Right click the Powershell icon and click Run as Administrator
6. Type: cd ”C:\Users\YOUR USERNAME\PhotonFirst\”
7. To display the installed python version on your system type: python -VV
8. Based on the python version, the following command should be entered:

Python version	Command
Python 3.8 (32 bits)	pip install p1_python_api-1.1.1-cp38-cp38-win32.whl
Python 3.8 (64 bits)	pip install p1_python_api-1.1.1-cp38-cp38-win_amd64.whl
Python 3.9 (32 bits)	pip install p1_python_api-1.1.1-cp39-cp39-win32.whl
Python 3.9 (64 bits)	pip install p1_python_api-1.1.1-cp39-cp39-win_amd64.whl
Python 3.10 (32 bits)	pip install p1_python_api-1.1.1-cp310-cp310-win32.whl
Python 3.10 (64 bits)	pip install p1_python_api-1.1.1-cp310-cp310-win_amd64.whl
Python 3.11 (32 bits)	pip install p1_python_api-1.1.1-cp311-cp311-win32.whl
Python 3.11 (64 bits)	pip install p1_python_api-1.1.1-cp311-cp311-win_amd64.whl

The PhotonFirst Python API will now be installed on you system.

3.2.2 Sample application - Modular Photonics Interrogator

This section describes how to configure the sample application in order to communicate with the connected Modular Photonics Interrogator.

1. Open mp-sampleapplication.py with a text editor
2. Change values for the following variables:

Variable	Value
MP_IP	IP address of the Modular Photonics Interrogator. This value can be retrieved using the FindMyMP application.
MP_SN	Serial number of the Modular Photonics Interrogator. This value can be retrieved using the FindMyMP application.
MP_CREDENTIALS	Username and password used for authenticating. If not changed, the default values can be used here.

3. Save the changes to the file

3.2.3 Sample application - (Switched)Gator

No configuration changes to the sample application are required: The sample application will automatically detect connected (Switched)Gators connected via USB.

4 Operation

This section describes how to interact with specific interrogators using the PhotonFirst API. Please make sure that the all required software is properly installed and configured as described in section 3 prior to using the API.

4.1 Modular Photonics Interrogator

The PhotonFirst API contains various functionalities related to configuration and data streaming for the Modular Photonics Interrogator. This section describes how to use the API and run the provided sample application.

4.1.1 Importing and instantiating the MpApi

Once the API has been installed the MpApi object can be imported in your source file:

```
from photonfirst.modular_photonics import MpApi
```

The MpApi object will allow you to interface with a specific MP:

```
MpApi(ip_address, serial_number, username, password, logger=None, frame_queue_size=-1)
Class which offers various methods for interfacing with the Modular Photonics Platform

@param ip_address The IP address of the MP
@param serial_number The serial_number of the MP
@param username The username
@param password The password
@param logger The logger to use (default=None). Must be of type logging.logger
@param frame_queue_size Limit the mp_data frame buffer
       (results in frame dropping but assures low latency)
@return MpApi object
```

4.1.2 MpApi member functions

```
start_measurement_datastream(callback, timeout=5)
Starts a measurement datastream

@param callback The callback method
@param timeout The maximum amount of time allowed for setting up the datastream
@return None
```

```
stop_measurement_datastream()
Stops a running measurement datastream
@return None
```

```
get_ffactor(timeout=10, verify_actual=True)
Gets the F-Factor value from the MP

@param timeout The maximum amount of time allowed for the request
@param verify_actual Keep waiting until the value received can be verified as actual
@return F-Factor as floating point number
```

```
set_ffactor(value, timeout=10)
Sets the F-Factor value

@param value The new value (integer value)
@param timeout The maximum amount of time allowed for the request
@return None
```

```
get_samplefrequency(timeout=10, verify_actual=True)
Gets the sample frequency value from the MP

@param timeout The maximum amount of time allowed for the request
@param verify_actual Keep waiting until the value received can be verified as actual
@return Sample frequency in Hz as integer
```

```
set_samplefrequency(value, timeout=10)
Sets the sample frequency

@param value The new value (integer value: 1000, 5000, 10000 or 19230)
@param timeout The maximum amount of time allowed for the request
@return None
```

```
get_cogthreshold(timeout=10, verify_actual=True)
Gets the CoG threshold value from the MP

@param timeout The maximum amount of time allowed for the request
@param verify_actual Keep waiting until the value received can be verified as actual
@return CoG threshold as integer
```

```
set_cogthreshold(value, timeout=10)
Sets the CoG threshold

@param value The new value (integer value)
@param timeout The maximum amount of time allowed for the request
@return None
```

```
get_optical_channel_sequence(timeout=10, verify_actual=True)
Gets the optical channel sequence from the MP

@param timeout The maximum amount of time allowed for the request
@param verify_actual Keep waiting until the value received can be verified as actual
@return List of values between 1-16
```

```
set_optical_channel_sequence(self, value, timeout=10)
Sets the optical channel sequence from the MP

@param value List of values between 1-16
@param timeout The maximum amount of time allowed for the request
@return None
```

```
get_optical_channel_hold_time(self, timeout=10, verify_actual=True)
Gets the number of samples retained for each channel.

@param timeout The maximum amount of time allowed for the request
@param verify_actual Keep waiting until the value received can be verified as actual
@return Integer value between 0-1000 (samples)
```

```
set_optical_channel_hold_time(self, value, timeout=10)
Sets the number of samples retained for each channel.

@param value Integer value between 0-1000 (samples)
@param timeout The maximum amount of time allowed for the request
@return None
```

```
get_version()
Gets the version number of this python library
@return Semantic version number
```

4.1.3 Running the sample application

After installation of the API as described in section 3, the sample code can be found in the PhotonFirst folder on the Windows Desktop. The sample code demonstrates how to use the functionality exposed by the API.

Prior to running the sample application, please make sure that the `MP_IP`, `MP_SN` and `MP_CREDENTIALS` are set to the correct values as described in section 3.2.2.

The following steps should be performed to run the sample application:

1. Click the `Start` menu button in windows and type `Powershell`
2. Right click the `Powershell` icon and click `Run as Administrator`
3. Type: `cd ”C:\Users\`**`YOUR USERNAME`**`\PhotonFirst\”`
4. Type: `python.exe mp-sampleapplication.py`

The sample application will now start and display the API version followed by a menu of options. Please note that the sample application will create a photonfirst-api.log file which can be used for debugging purposes.

4.2 (Switched)Gator Interrogator

The PhotonFirst API contains various functionalities related to configuration and data streaming for the Gator and SwitchedGator interrogators. This section describes how to use the API and run the provided sample application.

4.2.1 Importing and instantiating the GatorAPI

Once the API has been installed the GatorAPI object can be imported in your source file:

```
from photonfirst.gator_api import GatorAPI
```

The GatorApi object will allow you to interface with a specific (Switched)Gator:

```
GatorApi(logger=None)
Class which offers various methods for interfacing with the (Switched)Gator

@param logger The logger to use (default=None). Must be of type logging.logger
@return GatorApi object
```

After instantiating the GatorApi object, the connected (Switched)Gators can be resolved:

```
# Initialize the api by instantiating the GatorApi class.
api = GatorApi(None)
gators = api.get_gators()
```

The get_gators() method will provide a list of gators that could be resolved. The individual items in this list contain methods which can be invoked for interaction with a specific gator (see 4.2.2).

```
get_gators()
Gets a list of connected gators

@return List of P1Interface objects
```

4.2.2 P1Interface member functions

```
start_measurement_datastream(callback)
Starts a measurement datastream

@param callback The callback method
```

```
stop_measurement_datastream()
Stops a running measurement datastream
```

```
get_ffactor()
Gets the F-Factor value from the switchedgator

@return F-Factor as floating point number
```

```
set_ffactor(value)
Sets the F-Factor value

@param value The new value (integer value)
```

```
get_samplefrequency()
Gets the sample frequency value from the switchedgator

@return Sample frequency in Hz as integer
```

```
set_samplefrequency(value)
Sets the sample frequency

@param value The new value (integer value: 1000, 5000, 10000 or 19230)
```

```
get_cogthreshold()
Gets the CoG threshold value from the switchedgator

@return CoG threshold as integer
```

```
set_cogthreshold(value)
Sets the CoG threshold
@param value The new value (integer value)
```

```
get_raw_adas_values()
Gets the raw ADAS values
NOTE: This will fetch the values for the current channel (if switching enabled)
@return List with 64 unsigned integer values
```

```
get_dcc_adas_values()
Gets the dark current corrected ADAS values
NOTE: This will fetch the values for the current channel (if switching enabled)
@return List with 64 unsigned integer values
```

```
set_channel_switch_range(start_channel, end_channel)
Sets the channel switch range
@param start_channel The start channel (1-8)
@param end_channel The end channel (1-8)
```

```
set_averaging(enabled)
Enable averaging of output data between switches
@param enabled True if averaging should be enabled
```

4.2.3 Running the sample application

After installation of the API as described in section 3, the sample code can be found in the PhotonFirst folder on the Windows Desktop. The sample code demonstrates how to use the functionality exposed by the API. The following steps should be performed to run the sample application:

1. Click the `Start` menu button in windows and type `Powershell`
2. Right click the `Powershell` icon and click `Run as Administrator`
3. Type: `cd "C:\Users\`**`YOUR USERNAME`**`\PhotonFirst\"`
4. Type: `python.exe gator-sampleapplication.py`

The sample application will now start and display the API version followed by a menu of options. Please note that the sample application will create a photonfirst-api.log file which can be used for debugging purposes.